Models, Features and Outputs

Init setup

```
library(dplyr)
library(ggplot2)
library(readr)
library(caret)
library(pROC)
library(glmnet)
library(randomForest)
```

Load and preprocess data

```
df <- read_csv('acccident_data.csv')
# df_2021<-df %>% filter(CRASH_YEAR == 2021, !is.na(FATAL), !is.na(HOUR_OF_DAY))
df_2022 <- df %>%
  filter(CRASH_YEAR == 2022, !is.na(FATAL), !is.na(HOUR_OF_DAY)) %>%
  mutate(
    FATAL_OR_MAJ_INJ = factor(as.integer(FATAL_OR_MAJ_INJ)),
    MUNICIPALITY = factor(MUNICIPALITY, levels = unique(df$MUNICIPALITY)),
    INTERSECT_TYPE = factor(INTERSECT_TYPE, levels = c("00", "01", "02", "03", "04", "05",
                                                       "06", "07", "08", "09", "10", "99",
                                                       "11", "12", "13")),
    ROAD_CONDITION = factor(ROAD_CONDITION, levels = c(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 98, 99),
                            labels = c("Dry", "Wet", "Sand/mud/dirt/oil/gravel",
                                       "Snow covered", "Slush", "Ice", "Ice Patches",
                                       "Water - standing or moving", "Other",
                                       "Unknown (expired)", "Other", "Unknown")),
    ILLUMINATION = factor(ILLUMINATION, levels = c(1, 2, 3, 4, 5, 6, 8, 9),
                          labels = c("Daylight", "Dark - no street lights",
                                     "Dark - street lights", "Dusk", "Dawn",
                                     "Dark - unknown roadway lighting", "Other", "Other")),
    WEATHER = factor(WEATHER, levels = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 98, 99),
                     labels = c("Blowing Sand, Soil, Dirt", "Blowing Snow"
                                , "Clear", "Cloudy", "Fog, Smog, Smoke",
                                "Freezing Rain or Freezing Drizzle",
                                "Rain", "Severe Crosswinds", "Sleet or Hail", "Snow",
                                "Other", "Unknown"))
  ) %>%
  droplevels() # Ensure all levels are known and used
original_features <- c("FATAL_OR_MAJ_INJ","MUNICIPALITY",
                       "ROAD_CONDITION","LOCATION_TYPE"
                       ,"SPEED_LIMIT","INJURY_OR_FATAL",
```

```
                        "UNBELTED","ILLUMINATION",
                        "CURVED_ROAD","IMPAIRED_DRIVER",
                        "ROAD_OWNER",
                        "DISTRACTED","AGGRESSIVE_DRIVING","WEATHER","SCHOOL_ZONE")

additional_features <- c(original_features,"PED_COUNT","PERSON_COUNT","HIT_PARKED_VEHICLE")

df_2022_orig <- df_2022[original_features]
df_2022_add <- df_2022[additional_features]
# df_2022<-df_2022[c("FATAL_OR_MAJ_INJ","MUNICIPALITY","DAY_OF_WEEK","HOUR_OF_DAY",
#                     "ROAD_CONDITION","LOCATION_TYPE",
#                     "SPEED_LIMIT","INJURY_OR_FATAL","INTERSECTION",
#                     "TAILGATING","SPEEDING","LANE_COUNT",
# "MOTORCYCLE_COUNT",
# # "UNBELTED","ILLUMINATION","CURVED_ROAD","IMPAIRED_DRIVER"
# ,"ALCOHOL_RELATED","RELATION_TO_ROAD","ROAD_OWNER","DISTRACTED",
# "AGGRESSIVE_DRIVING","INTERSECT_TYPE","SMALL_TRUCK_COUNT"
# ,"AUTOMOBILE_COUNT")]
# # #"POLICE_AGCY","SCHOOL_ZONE",,,,"HIT_PARKED_VEHICLE",
# "TCD_TYPE",,,"PED_COUNT","PERSON_COUNT"
#                     # ,"UNBELTED_OCC_COUNT",
```

Splitting 70-30 train test Additional

```
set.seed(123)
trainingIndex_add <- createDataPartition(df_2022_add$INJURY_OR_FATAL,
                                         p = 0.6, list = FALSE)
trainingData_add <- df_2022_add[trainingIndex_add,]
testingData_add <- df_2022_add[-trainingIndex_add,]
trainingData_add<- trainingData_add %>% filter(is.na(SPEED_LIMIT)==FALSE
                                               & is.na(INJURY_OR_FATAL)==FALSE  )
testingData_add<- testingData_add %>% filter(is.na(SPEED_LIMIT)==FALSE
                                             & is.na(INJURY_OR_FATAL)==FALSE )
```

Splitting 70-30 train test Original features

```
set.seed(123)
trainingIndex_orig <- createDataPartition(df_2022_orig$INJURY_OR_FATAL,
                                          p = 0.7, list = FALSE)
trainingData_orig <- df_2022_orig[trainingIndex_orig,]
testingData_orig <- df_2022_orig[-trainingIndex_orig,]
trainingData_orig<- trainingData_orig %>% filter(is.na(SPEED_LIMIT)==FALSE &
                                                   is.na(INJURY_OR_FATAL)==FALSE  )
testingData_orig<- testingData_orig %>% filter(is.na(SPEED_LIMIT)==FALSE &
                                                 is.na(INJURY_OR_FATAL)==FALSE )
```

Create matrices Original features

```
testingData_orig$MUNICIPALITY <- factor(testingData_orig$MUNICIPALITY,
                                        levels = levels(trainingData_orig$MUNICIPALITY))

X_train_orig <- model.matrix(~ . - 1, data = trainingData_orig %>%
                               select(-INJURY_OR_FATAL, -FATAL_OR_MAJ_INJ))
Y_train_injury_orig <- as.numeric(trainingData_orig$INJURY_OR_FATAL) - 1
Y_train_fatal_orig <- as.numeric(trainingData_orig$FATAL_OR_MAJ_INJ) - 1
X_test_orig <- model.matrix(~ . - 1, data = testingData_orig %>%
                              select(-INJURY_OR_FATAL, -FATAL_OR_MAJ_INJ))
```

Create matrices Additional features

```
testingData_add$MUNICIPALITY <- factor(testingData_add$MUNICIPALITY,
                                       levels = levels(trainingData_add$MUNICIPALITY))

X_train_add <- model.matrix(~ . - 1, data = trainingData_add
                            %>% select(-INJURY_OR_FATAL, -FATAL_OR_MAJ_INJ))
Y_train_injury_add <- as.numeric(trainingData_add$INJURY_OR_FATAL) - 1
Y_train_fatal_add <- as.numeric(trainingData_add$FATAL_OR_MAJ_INJ) - 1
X_test_add <- model.matrix(~ . - 1, data = testingData_add
                           %>% select(-INJURY_OR_FATAL, -FATAL_OR_MAJ_INJ))
```

Ensure all categories are across both splits

```
X_train_df_orig <- as.data.frame(X_train_orig)
X_test_df_orig <- as.data.frame(X_test_orig)

common_columns <- intersect(colnames(X_train_df_orig), colnames(X_test_df_orig))
X_train_df_orig <- X_train_df_orig[, common_columns]
X_test_df_orig <- X_test_df_orig[, common_columns]

X_train_orig <- as.matrix(X_train_df_orig)
X_test_orig <- as.matrix(X_test_df_orig)


X_train_df_add <- as.data.frame(X_train_add)
X_test_df_add <- as.data.frame(X_test_add)

# selecting only the common columns between X_train_df
# and X_test_df because of diff in levels of testing and train
common_columns <- intersect(colnames(X_train_df_add), colnames(X_test_df_add))
X_train_df_add <- X_train_df_add[, common_columns]
X_test_df_add <- X_test_df_add[, common_columns]

X_train_add <- as.matrix(X_train_df_add)
X_test_add <- as.matrix(X_test_df_add)
#####
```

Original Model

FOR INJURY_OR_FATAL

Fit Lasso and Ridge Regression Models

```
lasso_model_injury_orig <- cv.glmnet(X_train_orig, Y_train_injury_orig,
                                     family = "binomial", alpha = 1)
ridge_model_injury_orig <- cv.glmnet(X_train_orig, Y_train_injury_orig,
                                     family = "binomial", alpha = 0)

# Predict and calculate AUC for INJURY_OR_FATAL
lasso_pred_injury_orig <- predict(lasso_model_injury_orig,
                                  newx = X_test_orig, type = "response",
                                  s = "lambda.min")
ridge_pred_injury_orig <- predict(ridge_model_injury_orig,
                                  newx = X_test_orig, type = "response",
                                  s = "lambda.min")

roc_lasso_injury_orig <- roc(testingData_orig$INJURY_OR_FATAL,
                             lasso_pred_injury_orig)
roc_ridge_injury_orig <- roc(testingData_orig$INJURY_OR_FATAL,
                             ridge_pred_injury_orig)

auc_lasso_injury_orig <- auc(roc_lasso_injury_orig)
auc_ridge_injury_orig <- auc(roc_ridge_injury_orig)

cat("AUC for INJURY_OR_FATAL ORIGINAL MODEL - Lasso:",
    auc_lasso_injury_orig, "\n")
```

```
## AUC for INJURY_OR_FATAL ORIGINAL MODEL - Lasso: 0.6194139
```

```
cat("AUC for INJURY_OR_FATAL ORIGINAL MODEL - Ridge:",
    auc_ridge_injury_orig, "\n")
```

```
## AUC for INJURY_OR_FATAL ORIGINAL MODEL - Ridge: 0.6111723
```

FOR FATAL_OR_MAJ_INJ

Fit Lasso and Ridge Regression Models

```
lasso_model_fatal_orig <- cv.glmnet(X_train_orig, Y_train_fatal_orig,
                                    family = "binomial", alpha = 1)
ridge_model_fatal_orig <- cv.glmnet(X_train_orig, Y_train_fatal_orig,
                                    family = "binomial", alpha = 0)

# Predict and calculate AUC for INJURY_OR_FATAL
lasso_pred_fatal_orig <- predict(lasso_model_fatal_orig, newx = X_test_orig,
                                 type = "response", s = "lambda.min")
```

```
ridge_pred_fatal_orig <- predict(ridge_model_fatal_orig, newx = X_test_orig,
                                 type = "response", s = "lambda.min")

roc_lasso_fatal_orig <- roc(testingData_orig$FATAL_OR_MAJ_INJ,
                            lasso_pred_fatal_orig)
roc_ridge_fatal_orig <- roc(testingData_orig$FATAL_OR_MAJ_INJ,
                            ridge_pred_fatal_orig)

auc_lasso_fatal_orig <- auc(roc_lasso_fatal_orig)
auc_ridge_fatal_orig <- auc(roc_ridge_fatal_orig)

cat("AUC for FATAL_OR_MAJ_INJ - Lasso:", auc_lasso_fatal_orig, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ - Lasso: 0.6447979
```

```
cat("AUC for FATAL_OR_MAJ_INJ - Ridge:", auc_ridge_fatal_orig, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ - Ridge: 0.6563058
```

Random Forest (Best Pick)

```
# Random Forest model for INJURY_OR_FATAL

rf_model_injury_orig <- randomForest(as.factor(INJURY_OR_FATAL) ~ .,
                                     data = trainingData_orig %>% select(-MUNICIPALITY),
                                     ntree = 500, mtry = sqrt(ncol(trainingData_orig) - 1))

# Random Forest model for FATAL_OR_MAJ_INJ
rf_model_fatal_orig <- randomForest(as.factor(FATAL_OR_MAJ_INJ) ~ .,
                                    data = trainingData_orig %>%
                                      select(-MUNICIPALITY), ntree = 500,
                                    mtry = sqrt(ncol(trainingData_orig) - 1))

pred_injury_rf_orig <- predict(rf_model_injury_orig,
                               newdata = testingData_orig %>%
                                 select(-MUNICIPALITY), type = "prob")[,2]
pred_fatal_rf_orig <- predict(rf_model_fatal_orig,
                              newdata = testingData_orig %>%
                                select(-MUNICIPALITY), type = "prob")[,2]

roc_rf_injury_orig <- roc(as.numeric(testingData_orig$INJURY_OR_FATAL) - 1,
                          pred_injury_rf_orig)
auc_rf_injury_orig <- auc(roc_rf_injury_orig)

roc_rf_fatal_orig <- roc(as.numeric(testingData_orig$FATAL_OR_MAJ_INJ) - 1,
                         pred_fatal_rf_orig)
auc_rf_fatal_orig <- auc(roc_rf_fatal_orig)

cat("AUC for INJURY_OR_FATAL ORIGINAL MODEL - Random Forest:",
    auc_rf_injury_orig, "\n")
```

```
## AUC for INJURY_OR_FATAL ORIGINAL MODEL - Random Forest: 0.5807069
```

```
cat("AUC for FATAL_OR_MAJ_INJ ORIGINAL MODEL - Random Forest:",
    auc_rf_fatal_orig, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ ORIGINAL MODEL - Random Forest: 0.721561
```

Additional Model

FOR INJURY_OR_FATAL

Fit Lasso and Ridge Regression Models

```
lasso_model_injury_add <- cv.glmnet(X_train_add, Y_train_injury_add,
                                    family = "binomial", alpha = 1)
ridge_model_injury_add <- cv.glmnet(X_train_add, Y_train_injury_add,
                                    family = "binomial", alpha = 0)

# Predictand calculate AUC for INJURY_OR_FATAL
lasso_pred_injury_add <- predict(lasso_model_injury_add,
                                 newx = X_test_add, type = "response",
                                 s = "lambda.min")
ridge_pred_injury_add <- predict(ridge_model_injury_add,
                                 newx = X_test_add, type = "response",
                                 s = "lambda.min")

roc_lasso_injury_add <- roc(testingData_add$INJURY_OR_FATAL,
                            lasso_pred_injury_add)
roc_ridge_injury_add <- roc(testingData_add$INJURY_OR_FATAL,
                            ridge_pred_injury_add)

auc_lasso_injury_add <- auc(roc_lasso_injury_add)
auc_ridge_injury_add <- auc(roc_ridge_injury_add)

cat("AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Lasso:",
    auc_lasso_injury_add, "\n")
```

```
## AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Lasso: 0.6746224
```

```
cat("AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Ridge:",
    auc_ridge_injury_add, "\n")
```

```
## AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Ridge: 0.6671791
```

FOR FATAL_OR_MAJ_INJ

Fit Lasso and Ridge Regression Models

```
lasso_model_fatal_add <- cv.glmnet(X_train_add, Y_train_fatal_add,
                                   family = "binomial", alpha = 1)
ridge_model_fatal_add <- cv.glmnet(X_train_add, Y_train_fatal_add,
                                   family = "binomial", alpha = 0)

# Predicting and calculate AUC for INJURY_OR_FATAL
lasso_pred_fatal_add <- predict(lasso_model_fatal_add,
                                newx = X_test_add, type = "response",
                                s = "lambda.min")
ridge_pred_fatal_add <- predict(ridge_model_fatal_add,
                                newx = X_test_add, type = "response",
                                s = "lambda.min")

roc_lasso_fatal_add <- roc(testingData_add$FATAL_OR_MAJ_INJ,
                           lasso_pred_fatal_add)
roc_ridge_fatal_add <- roc(testingData_add$FATAL_OR_MAJ_INJ, ridge_pred_fatal_add)

auc_lasso_fatal_add <- auc(roc_lasso_fatal_add)
auc_ridge_fatal_add <- auc(roc_ridge_fatal_add)

cat("AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL- Lasso:",
    auc_lasso_fatal_add, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL- Lasso: 0.7216194
```

```
cat("AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL - Ridge:",
    auc_ridge_fatal_add, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL - Ridge: 0.6909439
```

Random Forest (Best Pick)

```
# Random Forest model for INJURY_OR_FATAL

rf_model_injury_add <- randomForest(as.factor(INJURY_OR_FATAL) ~ .,
                                    data = trainingData_add %>%
                                      select(-MUNICIPALITY), ntree = 500,
                                    mtry = sqrt(ncol(trainingData_add) - 1))


rf_model_fatal_add <- randomForest(as.factor(FATAL_OR_MAJ_INJ) ~ .,
                                   data = trainingData_add %>%
                                     select(-MUNICIPALITY), ntree = 500,
                                   mtry = sqrt(ncol(trainingData_add) - 1))

pred_injury_rf_add <- predict(rf_model_injury_add,
                              newdata = testingData_add %>%
                                select(-MUNICIPALITY), type = "prob")[,2]
pred_fatal_rf_add <- predict(rf_model_fatal_add,
                             newdata = testingData_add %>%
```

```
                              select(-MUNICIPALITY), type = "prob")[,2]

roc_rf_injury_add <- roc(as.numeric(testingData_add$INJURY_OR_FATAL) - 1,
                         pred_injury_rf_add)
auc_rf_injury_add <- auc(roc_rf_injury_add)

roc_rf_fatal_add <- roc(as.numeric(testingData_add$FATAL_OR_MAJ_INJ) - 1,
                        pred_fatal_rf_add)
auc_rf_fatal_add <- auc(roc_rf_fatal_add)

cat("AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Random Forest:",
    auc_rf_injury_add, "\n")
```

```
## AUC for INJURY_OR_FATAL ADDITIONAL MODEL - Random Forest: 0.6454004
```

```
cat("AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL - Random Forest:",
    auc_rf_fatal_add, "\n")
```

```
## AUC for FATAL_OR_MAJ_INJ ADDITIONAL MODEL - Random Forest: 0.794442
```